論文の概要

１　申 請 者

防衛大学校　グェン　ダット　トゥオン

２　論文題目

GPU を用いたハッシュ関数 Keccak の高速化に関する研究

３　論文の概要

現在，ネットバンキングや，電子カルテなど，個人情報や秘匿性の高い情報を扱う機会が増加している．これらの情報を保護するためには，暗号化技術やハッシュ関数を利用する必要がある．ハッシュ関数の安全性については，第 1 原像計算困難性，第 2 原像計算困難性及び衝突困難性の 3 つの特徴で担保されている．本研究では，ハッシュ関数を用いたパスワード管理の安全性について考察するため，主としてハッシュ関数の第 1 原像計算困難性に対する安全性評価に関する考察を進めることとした．与えられたハッシュ値を出力とするメッセージを見つけることは，計算量的に困難である．しかし，ハッシュ関数を高速化実装することにより，短い時間内で多くの計算ができ，計算量的に困難であった問題を解決してしまう可能性がある．つまり，高速化実装により処理時間を減らすほど，ハッシュ関数の安全性はより低下することになる．

本研究では，GPU 向けの統合開発環境 CUDA を用いて，ハッシュ関数 SHA-3 の原案となった Keccak について高速化実装を行い，処理速度の測定結果から安全性について考察する．ハッシュ関数の実装において，1 つのパスワードを GPU 上の 1 スレッドを用いてハッシュ化処理を行ない，同時に多数のスレッドを起動して並列処理した．その実装において，次の 4 つの高速化のための方法を提案する．1 つ目はルックアップテーブルの再構成，2 つ目は GPU 上のメモリ階層の有効利用と，定数の適切なメモリへの配置である．3 つ目はブロック数・スレッド数の変更，最後の 4 つ目は GPU でのストリームを用いて，データ転送と計算のオーバーラップを行う．これらの手法を用いて GeForce GTX 1080 に実装した結果，最大 64.582 GB/s のスループットを実現できた．パスワードクラッキングツールである Hashcat を同じ環境で実行し，実行時間を測定した結果，最大一秒あたり 769.6 メガハッシュを処理できた．一方，本研究のスループットは一秒あたり 909.6 メガハッシュを処理できることを示してる．本研究の結果を用いて Hashcat のようにパスワードに対する総当たり攻撃の性能を向上することが可能なことを示した．

また，パスワードクラッキングの代表的な方法として，総当たり攻撃，辞書攻撃，そしてレインボーテーブルが存在する．総当たり攻撃では事前計算を必要とせず，メモリの消費も少ない一方，計算時間が掛かるという問題がある．辞書攻撃では，事前にパスワード候補を辞書に登録するため，その辞書を保存するために多くのメモリの確保が必要となる．この 2 つの攻撃方法の欠点を改善したものがレインボーテーブル攻撃であり，辞書攻撃と同じように事前にデ

ータを準備するため，総当たり攻撃よりも時間が掛からず，特殊な計算方法により辞書攻撃よりも少ないメモリで実現できる攻撃方法である．しかし，使用する文字などのパスワード候補の対象を変更した場合，必ずレインボーテーブルを再生成する必要があり，この準備には大きく時間が掛かる．レインボーテーブルは，事前計算テーブルの一種であり，適当に決めたパスワード候補からあらかじめハッシュ値を計算しておく．そして，そのハッシュ値からパスワード候補を生成する還元関数を用いて，別のパスワード候補を生成する。そして，ハッシュ化とパスワード候補の生成を繰り返し，この一連の処理で生成されるパスワード候補の集まりをチェーンと呼ぶ．このチェーンは，最初のパスワード候補である Starting Point（SP）と，最後のパスワード候補である Endpoint（EP）のペアのみを保存しており，これにより，使用するメモリを削減している．そして，このチェーンを集めたものがレインボーテーブルになる．

本研究では，1 つのチェーンをの生成を GPU の 1 スレッドに割り当て，生成時間の高速化を図った．生成時間を短縮するために，チェーン内のハッシュ化処理の部分において，ハッシュ関数 Keccak の高速化実装を活用した．チェーンの生成処理においては，ハッシュ値と還元関数を用いてパスワード候補を生成する．ここで生成されたパスワード候補は還元関数の性質により衝突が起きてしまうと，チェーン内のその後のパスワード候補が全て重複してしまい，レインボーテーブルの効率が落ちることになる．チェーン内や別のチェーンでこのパスワード候補の重複が発生しても次のチェーンの部分で重複しないように，還元関数にはパスワード候補のチェーン内の位置情報を追加した．これにより，チェーンの異なる位置におけるパスワード候補の重複を避けることができる．この提案した手法で生成されたレインボーテーブルを評価し，攻撃効果の予測・議論を行う．4 文字のパスワードを対象とした実装の結果，チェーンの長さが 50，かつチェーンの数が 4,598,517 の場合，生成されたレインボーテーブルのパスワード候補の網羅率は 98.55％であった．本手法を用いて CPU＋GPU を使用したレインボーテーブルの生成は CPU のみで生成する場合に比べると，約 239 倍高速化できた．

４　キーワード

ハッシュ関数，Keccak，レインボーテーブル，GPGPU，高速化